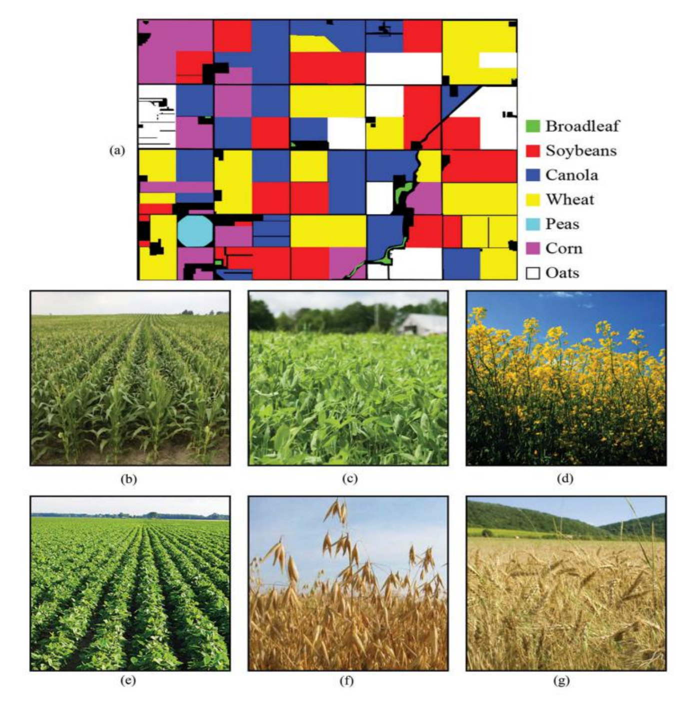
CROPLAND CLASSIFICATION USING OPTICAL AND RADAR DATA

TEAM: AKASH A (191EC102), NAVRATAN (191EC133) AND ROHAN J (191EC147)

MENTOR: DR. RAGHAVENDRA B S

AN OVERVIEW

- ❑ Motivation: Given radar and optical feature information, classify data of a single pixel into one of seven crop types
- ❑ Possibility of use by NGOs and governments for agriculture planning and management
- ❑ Efficient Classifier - Full image not needed; Only information about a single pixel is sufficient (Less Storage)
- ❑ Dataset Available At: https://archive.ics.uci.edu/ml/datasets/Crop+mapping+using+fused+optical-radar+data+set

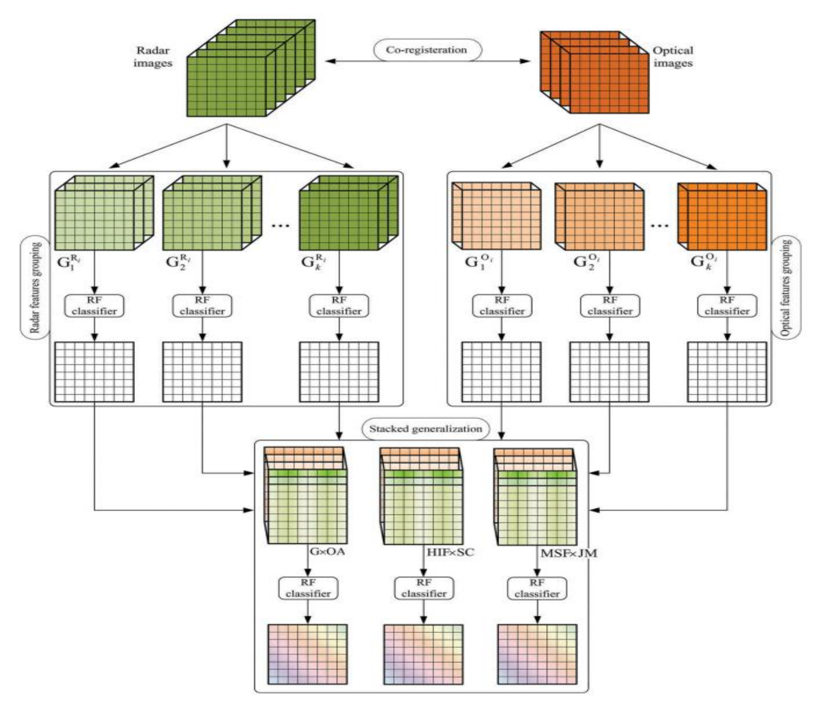


DESCRIPTION OF DATASET

- Supervised Learning Problem - Data consists of 174 features - optical and radar taken on two different dates
- Radar Features - Collected by sending radio waves in the L band and measuring characteristics of the reflected light
- Optical Features - Spectral data collected by sending out waves in the RGB, Near Infrared (NIR) and RedEye Spectrum
- Imbalanced Dataset - Seven kinds of crops; Number of samples for each crop was different

Class	Corn	Peas	Canola	Soybeans	Oats	Wheat	Broadleaf
No. of samples	39162	3598	75673	74067	47117	85074	1143

TECHNIQUES ADOPTED - BALANCED TRAINING



- ❑ Features were divided into groups based on their similarity to optimize performance
- ❑ A Random Forest Classifier was used to train data from each group and the accuracy was collected
- ❑ Each group of data was then multiplied by the overall accuracy and then stacked together
- ❑ Finally, training was done on modified data using another RF classifier

FEATURE GROUPS

RADAR FEATURES

OPTICAL FEATURES

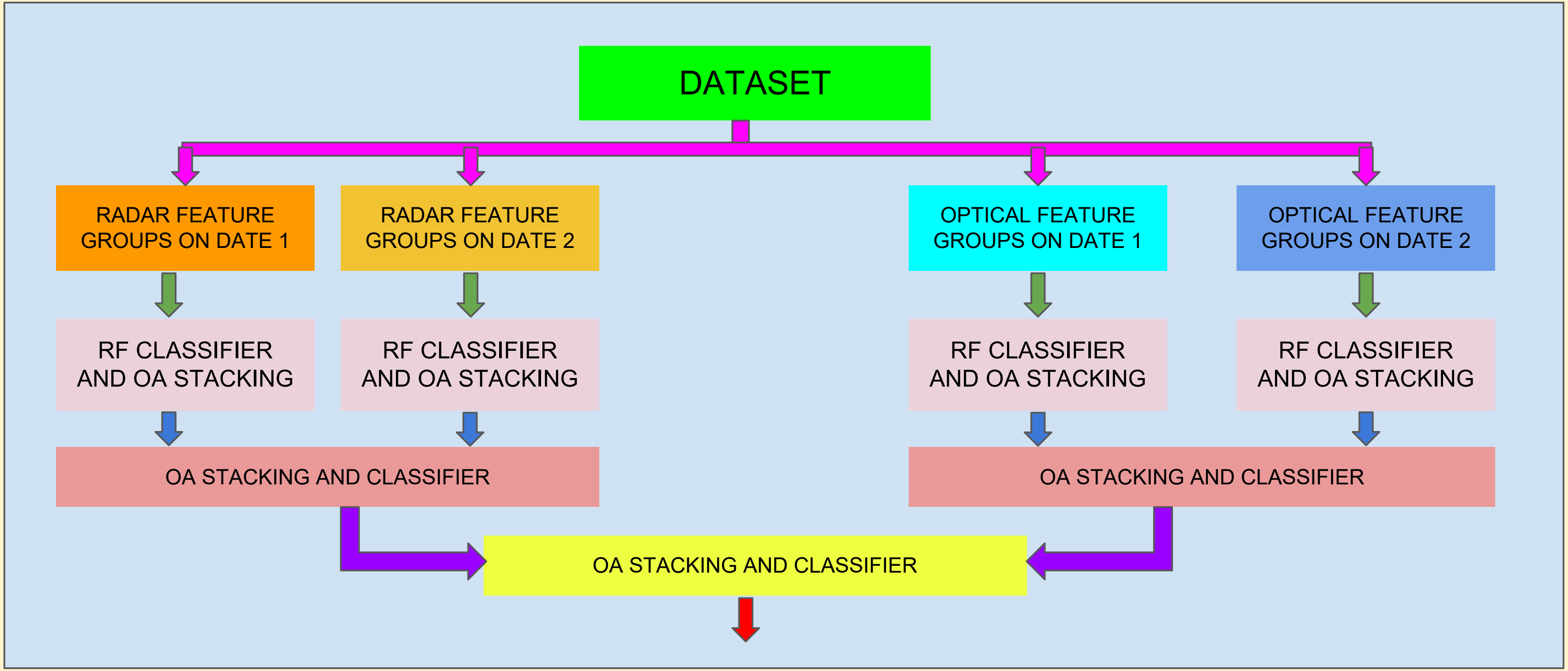
Features and formulas (G: groups, R: radar, i = 1 to 7)

$G_1^{R_i}$	$\sigma_{hh} = 10\log_{10}\lvert S_{hh}\rvert^2, \sigma_{hv} = 10\log_{10}\lvert S_{hv}\rvert^2, \sigma_{vv} = 10\log_{10}\lvert S_{vv}\rvert^2, \sigma_{rr} = 10\log_{10}\lvert S_{rr}\rvert^2, \sigma_{rl} = 10\log_{10}\lvert S_{rl}\rvert^2, \sigma_{ll} = 10\log_{10}\lvert S_{ll}\rvert^2$ {h : horizontal, v : vertical, r : right − handed, l : left − handed}
$G_2^{R_i}$	$R_{hhvv} = 10\log_{10}(\lvert S_{hh}\rvert^2/\lvert S_{vv}\rvert^2), R_{hvhh} = 10\log_{10}(\lvert S_{hv}\rvert^2/\lvert S_{hh}\rvert^2), R_{hvvv} = 10\log_{10}(\lvert S_{hv}\rvert^2/\lvert S_{vv}\rvert^2),$ $R_{rrll} = 10\log_{10}(\lvert S_{rr}\rvert^2/\lvert S_{ll}\rvert^2), R_{rlrr} = 10\log_{10}(\lvert S_{rl}\rvert^2/\lvert S_{rr}\rvert^2), R_{rlll} = 10\log_{10}(\lvert S_{rl}\rvert^2/\lvert S_{ll}\rvert^2)$
$G_3^{R_i}$	$R_{hh} = 10\log_{10}(\lvert S_{hh}\rvert^2/\text{span}), R_{hv} = 10\log_{10}(\lvert S_{hv}\rvert^2/\text{span}), R_{vv} = 10\log_{10}(\lvert S_{vv}\rvert^2/\text{span}),$ $R_{rr} = 10\log_{10}(\lvert S_{rr}\rvert^2/\text{span}), R_{rl} = 10\log_{10}(\lvert S_{rl}\rvert^2/\text{span}), R_{ll} = 10\log_{10}(\lvert S_{ll}\rvert^2/\text{span})$ $\{\text{span} = \lvert S_{hh}\rvert^2 + 2\lvert S_{hv}\rvert^2 + \lvert S_{vv}\rvert^2 = \lvert S_{rr}\rvert^2 + 2\lvert S_{rl}\rvert^2 + \lvert S_{ll}\rvert^2\}$
$G_4^{R_i}$	$\rho_{hhvv} = \left\lvert \frac{S_{hh}.S_{vv}^*}{\sqrt{(S_{hh}.S_{hh}^*)(S_{vv}.S_{vv}^*)}} \right\rvert, \rho_{hvhh} = \left\lvert \frac{S_{hv}.S_{hh}^*}{\sqrt{(S_{hv}.S_{hv}^*)(S_{hh}.S_{hh}^*)}} \right\rvert, \rho_{hvvv} = \left\lvert \frac{S_{hv}.S_{vv}^*}{\sqrt{(S_{hv}.S_{hv}^*)(S_{vv}.S_{vv}^*)}} \right\rvert;$ $\rho_{rrll} = \left\lvert \frac{S_{rr}.S_{ll}^*}{\sqrt{(S_{rr}.S_{rr}^*)(S_{ll}.S_{ll}^*)}} \right\rvert, \rho_{rlrr} = \left\lvert \frac{S_{rl}.S_{rr}^*}{\sqrt{(S_{rl}.S_{rl}^*)(S_{rr}.S_{rr}^*)}} \right\rvert, \rho_{rlll} = \left\lvert \frac{S_{rl}.S_{ll}^*}{\sqrt{(S_{rl}.S_{rl}^*)(S_{ll}.S_{ll}^*)}} \right\rvert$ {∗ : complex conjugate, . : vector dot product}
$G_5^{R_i}$	$\lambda_1, \lambda_2, \lambda_3, \quad H = -\sum_{i=1}^3 p_i \log_3 p_i, A = (\lambda_2 - \lambda_3)/(\lambda_2 + \lambda_3), \bar{\alpha} = \sum_{i=1}^3 p_i \alpha_i \quad \{p_i = \lambda_i / \sum_{k=1}^3 \lambda_k\}$ $HA, H(1-A), (1-H)A, (1-H)(1-A), \quad \psi = \min(\lambda_1, \lambda_2, \lambda_3)/(\lambda_1 + \lambda_2 + \lambda_3), \text{RVI} = 4\lambda_3/(\lambda_1 + \lambda_2 + \lambda_3)$
$G_6^{R_i}$	$\lvert\alpha\rvert^2 = \frac{\sqrt{2}}{2}\lvert S_{hh} + S_{vv}\rvert^2, \lvert\beta\rvert^2 = \frac{\sqrt{2}}{2}\lvert S_{hh} - S_{vv}\rvert^2, \lvert\gamma\rvert^2 = 2\lvert S_{hv}\rvert^2, \lvert k_s\rvert^2 = \lvert S_{rl}\rvert^2, \lvert k_d\rvert^2 = \min(\lvert S_{rr}\rvert^2, \lvert S_{ll}\rvert^2), \lvert k_h\rvert^2 = \text{abs}(\lvert S_{rr}\rvert^2 - \lvert S_{ll}\rvert^2)$ {s : surface scattering, d : double − bounce scattering, h : helix scattering}
$G_7^{R_i}$	$P_s = f_s(1 + \lvert\beta\rvert^2), P_d = f_d(1 + \lvert\alpha\rvert^2), P_v = f_v, P_s^Y = f_s(1 + \lvert\beta\rvert^2), P_d^Y = f_d(1 + \lvert\alpha\rvert^2), P_v^Y = f_v, P_c^Y = f_c$ {v : volume scattering, c : helix scattering}

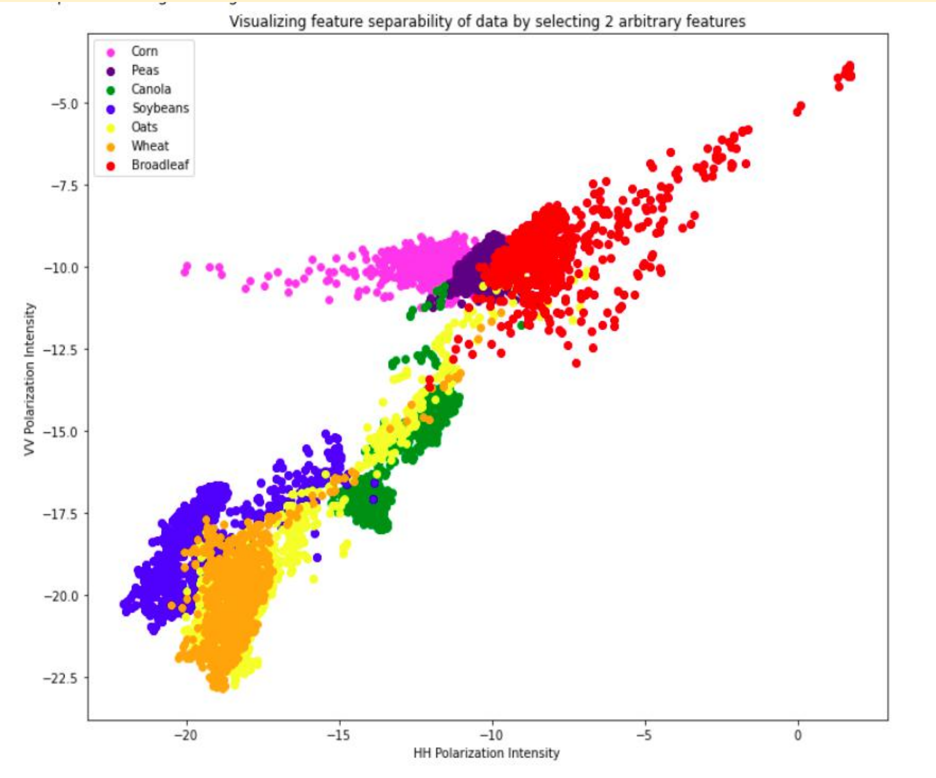
Features and formulas (G: groups, O: optical, i = 1 to 5)

$G_1^{O_i}$	R_B: 440 − 510nm, R_G: 520 − 590nm, R_R: 630 − 685nm, R_{RE}: 690 − 730nm, R_{NIR}: 760 − 850nm R: Reflectance
$G_2^{O_i}$	$\text{NDVI} = (R_{NIR} - R_R)/(R_{NIR} + R_R)$ $\text{SR} = R_{NIR}/R_R$ $\text{RGRI} = R_G/R_R$ $\text{EVI} = 2.5(R_{NIR} - R_R)/(R_{NIR} + 6R_R - 7.5R_B + 1)$ $\text{ARVI} = (R_{NIR} - (2R_R - R_B))/(R_{NIR} + (2R_R - R_B))$ $\text{SAVI} = (1 + 0.5)(R_{NIR} - R_R)/(R_{NIR} + R_R + 0.5)$ $\text{NDGI} = (R_G - R_R)/(R_G + R_R)$ $\text{gNDVI} = (R_{NIR} - R_G)/(R_{NIR} + R_G)$ $\text{MTVI2} = 1.5(1.2(R_{NIR} - R_G) - 2.5(R_R - R_G))/\sqrt{(2R_{NIR} + 1)^2 - (6R_{NIR} - 5\sqrt{R_R}) - 0.5}$
$G_3^{O_i}$	$\text{NDVIre} = (R_{NIR} - R_{RE})/(R_{NIR} + R_{RE})$ $\text{SRre} = R_{NIR}/R_{RE}$ $\text{NDGIre} = (R_G - R_{RE})/(R_G + R_{RE})$ $\text{RTVIcore} = 100(R_{NIR} - R_{RE}) - 10(R_{NIR} - R_G)$ $\text{RNDVI} = (R_{RE} - R_R)/(R_{RE} + R_R)$ $\text{TCARI} = 3((R_{RE} - R_R) - 0.2(R_{RE} - R_G)(R_{RE}/R_R))$ $\text{TVI} = 0.5(120(R_{RE} - R_G) - 200(R_R - R_G))$ $\text{PRI2} = R_{RE}/R_R$
$G_4^{O_i}$	$\mu_{PC1}, \sigma_{PC1}, \text{HOM}_{PC1}, \text{CON}_{PC1}, \text{DIS}_{PC1}, H_{PC1}, \text{ASM}_{PC1}, \text{COR}_{PC1}$ from GLCM of PC1
$G_5^{O_i}$	$\mu_{PC2}, \sigma_{PC2}, \text{HOM}_{PC2}, \text{CON}_{PC2}, \text{DIS}_{PC2}, H_{PC2}, \text{ASM}_{PC2}, \text{COR}_{PC2}$ from GLCM of PC2

IMPLEMENTATION



RESULTS OBTAINED EARLIER



STACKED FEATURES	OVERALL ACCURACY
RADAR FEATURES - DATE 1	60.33%
RADAR FEATURES - DATE 2	75.24%
RADAR FEATURES	70.04%
OPTICAL FEATURES - DATE 1	66.67%
OPTICAL FEATURES - DATE 2	66.28%
OPTICAL FEATURES	59.33%
RADAR & OPTICAL FEATURES	64.03%

FINAL RESULTS AFTER IMPROVISATION

- Earlier, predicted values were multiplied by overall accuracy and stacked for next stage - no improvement
- Alternative was tried by multiplying the feature values themselves with overall accuracy - significant improvement
- Final overall accuracy exceeded 90% - Exceeded performance obtained by research in given problem
- Hyperparameters for RF classifier - No. of decision tree and number of features given to each decision tree

STACKED FEATURES	OVERALL ACCURACY
RADAR FEATURES - DATE 1	60.42%
RADAR FEATURES - DATE 2	75.22%
RADAR FEATURES	86.87%
OPTICAL FEATURES - DATE 1	66.59%
OPTICAL FEATURES - DATE 2	65.86%
OPTICAL FEATURES	78.60%
RADAR & OPTICAL FEATURES	90.55%

IMBALANCED TRAINING - TECHNIQUES ADOPTED

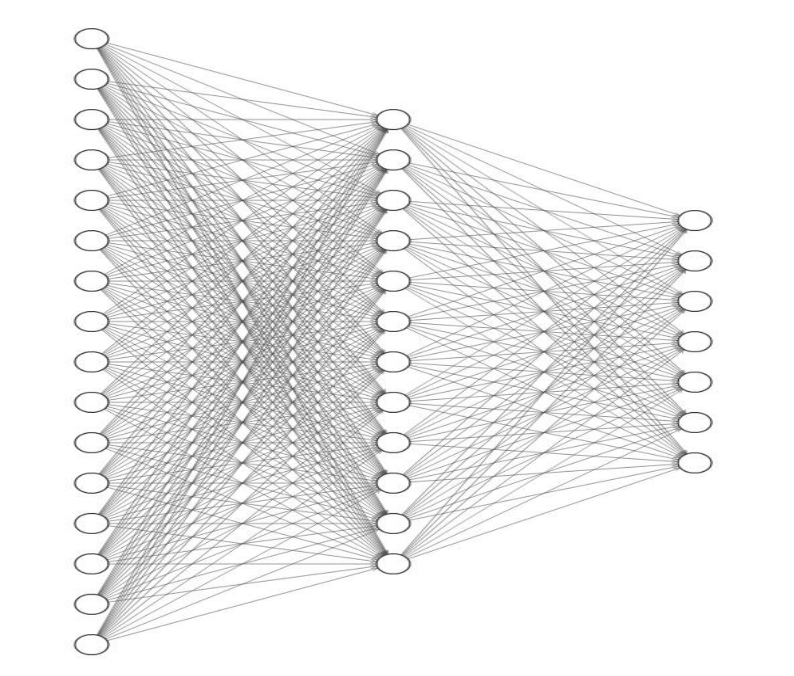
- Random Forest Classifier was tried out - Gave good results of about 96% for each of the groups itself but took too much time to converge. Hence, other techniques were explored

- Neural Network architectures were thought of as an alternative since they can implement complex non-linear classifiers very optimally

- Convolutional Networks could not be used since they have applications for images but here we are dealing with pixel data. Thus feedforward networks were resorted to with variation of hyperparameters

- Two-thirds of the data was arbitrarily assigned as the training set and the remaining one-third as the test set. This leads to an imbalanced training since the original dataset itself is imbalanced.

NEURAL NETWORK ARCHITECTURES



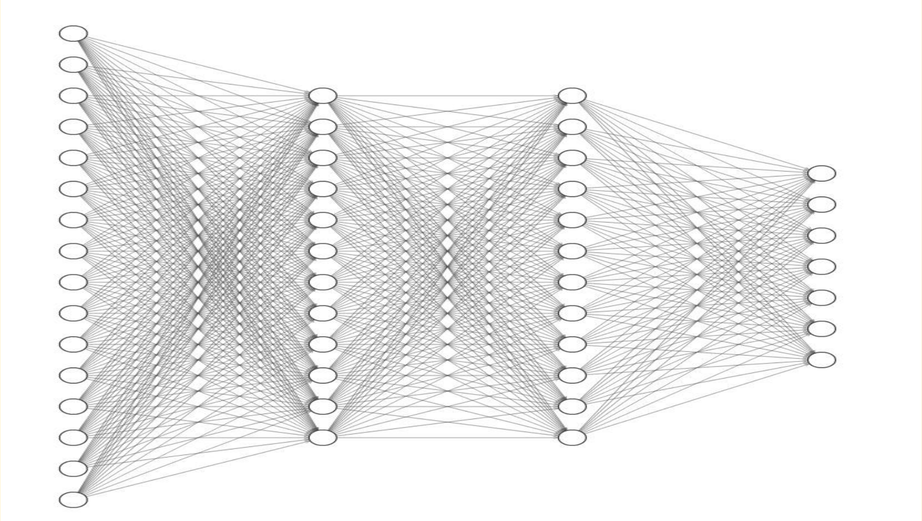
Single Hidden Layer

- Three layers - Input Layer, Output Layer and a Hidden Layer
- Input Layer has 174 neurons - equal to the number of input features
- Output Layer has 7 neurons - Each represents the probability of the feature vector belonging to one of seven classes
- Number of neurons in hidden layer is an important hyperparameter - geometric mean of input and output was considered here

NEURAL NETWORK ARCHITECTURES - CONTD.

- Number of hidden layers was chosen by method of trial and error - Hidden layers were added till accuracy no longer improved

- Adam optimizer was used for training with Categorical Cross Entropy as error metric



RESULTS

ONE HIDDEN LAYER

```
Epoch 15/500
500/500 [==============================] - 2s 4ms/step - loss: 9.1349 - accuracy: 0.9198 - val_loss: 2.4975 - val_accuracy: 0.9570
Epoch 16/500
500/500 [==============================] - 2s 3ms/step - loss: 12.1884 - accuracy: 0.9125 - val_loss: 3.1065 - val_accuracy: 0.9492
Epoch 17/500
500/500 [==============================] - 2s 3ms/step - loss: 8.2772 - accuracy: 0.9260 - val_loss: 4.5093 - val_accuracy: 0.9531
Epoch 18/500
500/500 [==============================] - 2s 4ms/step - loss: 9.5611 - accuracy: 0.9198 - val_loss: 3.2605 - val_accuracy: 0.9336
Epoch 19/500
500/500 [==============================] - 2s 3ms/step - loss: 8.3610 - accuracy: 0.9232 - val_loss: 5.2934 - val_accuracy: 0.9336
Epoch 20/500
500/500 [==============================] - 2s 3ms/step - loss: 10.1683 - accuracy: 0.9126 - val_loss: 3.1727 - val_accuracy: 0.9570
Epoch 21/500
500/500 [==============================] - 2s 3ms/step - loss: 9.5007 - accuracy: 0.9239 - val_loss: 44.7430 - val_accuracy: 0.7812
Epoch 22/500
500/500 [==============================] - 2s 4ms/step - loss: 12.0754 - accuracy: 0.9141 - val_loss: 5.0989 - val_accuracy: 0.9180
Epoch 23/500
500/500 [==============================] - 2s 3ms/step - loss: 6.6663 - accuracy: 0.9294 - val_loss: 7.5614 - val_accuracy: 0.9062
Epoch 24/500
500/500 [==============================] - 2s 3ms/step - loss: 10.2480 - accuracy: 0.9125 - val_loss: 9.5240 - val_accuracy: 0.8750
Epoch 25/500
497/500 [============================>.] - ETA: 0s - loss: 7.2080 - accuracy: 0.9291INFO:tensorflow:Assets written to: /SingleLayerModel/assets
500/500 [==============================] - 2s 5ms/step - loss: 7.1794 - accuracy: 0.9293 - val_loss: 0.2448 - val_accuracy: 0.9922
Epoch 26/500
500/500 [==============================] - 2s 3ms/step - loss: 7.9251 - accuracy: 0.9266 - val_loss: 1.0116 - val_accuracy: 0.9688
Epoch 27/500
500/500 [==============================] - 1s 3ms/step - loss: 7.6420 - accuracy: 0.9257 - val_loss: 5.1029 - val_accuracy: 0.9648
Epoch 28/500
```

TWO HIDDEN LAYERS

```
Epoch 15/500
497/500 [============================>.] - ETA: 0s - loss: 55.1457 - accuracy: 0.7330INFO:tensorflow:Assets written to: /SingleLayerModel/assets
500/500 [==============================] - 3s 6ms/step - loss: 54.8758 - accuracy: 0.7337 - val_loss: 11.5499 - val_accuracy: 0.8828
Epoch 16/500
500/500 [==============================] - 2s 3ms/step - loss: 33.3535 - accuracy: 0.7633 - val_loss: 18.2218 - val_accuracy: 0.8008
Epoch 17/500
500/500 [==============================] - 2s 3ms/step - loss: 38.6675 - accuracy: 0.7456 - val_loss: 43.4058 - val_accuracy: 0.5547
Epoch 18/500
491/500 [============================>.] - ETA: 0s - loss: 27.7190 - accuracy: 0.7846INFO:tensorflow:Assets written to: /SingleLayerModel/assets
500/500 [==============================] - 3s 6ms/step - loss: 27.6354 - accuracy: 0.7842 - val_loss: 9.0864 - val_accuracy: 0.9062
Epoch 19/500
500/500 [==============================] - 2s 3ms/step - loss: 40.0771 - accuracy: 0.7650 - val_loss: 13.1498 - val_accuracy: 0.7734
Epoch 20/500
500/500 [==============================] - 2s 3ms/step - loss: 20.9534 - accuracy: 0.8002 - val_loss: 9.0233 - val_accuracy: 0.8633
Epoch 21/500
500/500 [==============================] - 2s 3ms/step - loss: 27.1688 - accuracy: 0.7877 - val_loss: 73.0210 - val_accuracy: 0.5586
Epoch 22/500
500/500 [==============================] - 2s 4ms/step - loss: 25.5658 - accuracy: 0.7926 - val_loss: 61.9695 - val_accuracy: 0.6172
Epoch 23/500
500/500 [==============================] - 2s 3ms/step - loss: 24.9595 - accuracy: 0.7956 - val_loss: 23.7484 - val_accuracy: 0.8086
Epoch 24/500
500/500 [==============================] - 2s 3ms/step - loss: 22.6054 - accuracy: 0.8032 - val_loss: 10.8737 - val_accuracy: 0.8828
Epoch 25/500
500/500 [==============================] - 2s 4ms/step - loss: 21.9678 - accuracy: 0.8109 - val_loss: 9.3023 - val_accuracy: 0.8555
Epoch 26/500
500/500 [==============================] - 2s 3ms/step - loss: 26.8509 - accuracy: 0.7889 - val_loss: 18.1688 - val_accuracy: 0.7461
Epoch 27/500
500/500 [==============================] - 2s 3ms/step - loss: 23.8704 - accuracy: 0.7999 - val_loss: 10.0237 - val_accuracy: 0.8789
Epoch 28/500
```

RESULTS - CONTD.

```
Probability Values generated by the model :

[[0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 1.0000000e+00
  3.8196823e-24 0.0000000e+00]
 [0.0000000e+00 0.0000000e+00 2.8908889e-34 0.0000000e+00 9.9998927e-01
  1.0762273e-05 3.9544315e-37]
 [0.0000000e+00 0.0000000e+00 0.0000000e+00 1.0000000e+00 0.0000000e+00
  0.0000000e+00 0.0000000e+00]
 [0.0000000e+00 0.0000000e+00 1.0000000e+00 0.0000000e+00 0.0000000e+00
  0.0000000e+00 0.0000000e+00]
 [0.0000000e+00 0.0000000e+00 0.0000000e+00 1.0000000e+00 0.0000000e+00
  0.0000000e+00 0.0000000e+00]
 [0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00
  1.0000000e+00 0.0000000e+00]
 [1.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00
  0.0000000e+00 0.0000000e+00]
 [1.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00
  0.0000000e+00 0.0000000e+00]
 [0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00
  1.0000000e+00 0.0000000e+00]
 [0.0000000e+00 0.0000000e+00 0.0000000e+00 1.0000000e+00 0.0000000e+00
  0.0000000e+00 0.0000000e+00]]


Class Selected based on Maximum Probability :

[5 5 4 3 4 6 1 1 6 4]


Actual Class :

[5 6 4 3 4 6 1 1 6 4]
```

RESULTS - CONFUSION MATRICES (BALANCED)

		PREDICTED CLASS							
ACTUAL CLASS		CLASS 1	CLASS 2	CLASS 3	CLASS 4	CLASS 5	CLASS 6	CLASS 7	RECALL
	CLASS 1	37143	28	451	130	639	90	680	94.85%
	CLASS 2	0	3468	117	2	10	0	0	96.41%
	CLASS 3	7	163	75187	88	73	94	60	99.36%
	CLASS 4	298	22	5349	59827	549	8020	1	80.78%
	CLASS 5	137	0	164	57	40428	6274	56	85.81%
	CLASS 6	424	12	183	387	6416	77596	55	91.21%
	CLASS 7	0	0	0	0	0	0	1143	100%
	PRECISION	97.72%	93.91%	91.63%	98.90%	84.02%	84.28%	57.32%	

RESULTS - F1 SCORES (BALANCED)

CLASS 1	96.26%
CLASS 2	95.14%
CLASS 3	95.34%
CLASS 4	88.93%
CLASS 5	84.91%
CLASS 6	87.61%
CLASS 7	72.87%
MACRO AVG	88.72%

- All classes except class 7 have an F1 score value of greater than 80% which is interpreted to be quite good
- Macro average comes out to be 88.72% which again is interpreted as a quite accurate for an imbalanced multi-class classification
- Class 7 is observed to have perfect recall but a poor precision. Similarly, classes 5 and 6 are observed to be poorly separable
- To make further conclusions, it would be necessary to compare with recent research as these are relative values.

RESULTS - CONFUSION MATRICES (IMBALANCED)

ACTUAL CLASS	PREDICTED CLASS							
	CLASS 1	CLASS 2	CLASS 3	CLASS 4	CLASS 5	CLASS 6	CLASS 7	RECALL
CLASS 1	38639	30	79	328	8	24	53	98.67%
CLASS 2	27	3532	12	23	0	3	0	98.19%
CLASS 3	38	48	75246	218	87	33	2	99.44%
CLASS 4	486	132	124	72951	129	239	5	98.49%
CLASS 5	1482	4	85	355	42553	2486	150	90.32%
CLASS 6	241	0	107	281	3803	80539	102	94.67%
CLASS 7	235	2	18	4	5	25	854	74.72%
PRECISION	93.90%	94.24%	99.44%	98.37%	91.34%	96.63%	73.24%	

RESULTS - F1 SCORES (IMBALANCED)

CLASS 1	96.23%
CLASS 2	96.17%
CLASS 3	99.44%
CLASS 4	98.43%
CLASS 5	90.83%
CLASS 6	95.64%
CLASS 7	73.97%
MACRO AVG	92.96%

- Imbalanced training was observed to given a better distribution between precision and recall as compared to balanced training
- Particularly for class 7 with least number of samples, significant improvement in precision value by trading with recall (no free lunch)
- F1-Score found to improve significantly for most of the classes
- Conclusion: Imbalanced training gives better results as compared to balanced training

IMPROVEMENTS

- ❑ More advanced methods can be explored for stacking fusion strategies - Variable feature selection, Bhattacharya's distance based selection, etc.
- ❑ More advanced and standard architectures for training the neural networks could offer improved accuracy
- ❑ Study on how data was collected and how the radar and optical features were combined - Processes like coregistration and orthorectification

REFERENCES

- Iman Khosravi & Seyed Kazem Alavipanah (2019) A random forest-based framework for crop mapping using temporal, spectral, textural and polarimetric observations, International Journal of Remote Sensing, 40:18, 7221-7251, DOI: 10.1080/01431161.2019.1601285
- https://www.analyticsvidhya.com/blog/2021/04/getting-into-random-forest-algorithms/